Homework 2 Programming Problem 8 (30 points)

Problem Description

In this problem you will use linear least squares to fit a linear function to a 3D temperature field, with x,y,z locations and an associated temperature T.

Fill out the notebook as instructed, making the requested plots and printing necessary values.

You are welcome to use any of the code provided in the lecture activities.

Summary of deliverables:

Results:

- Predicted temperature `T(5,5,5)` using a hand-coded LLS squares model with a linear function
- Direction of travel from `(5,5,5)` to experience the greatest decrease in temperature

Discussion:

- Reasoning for how we can use our fitted function to determine the direction of greatest decrease in temperature

Imports and Utility Functions:

In [2]:
```
import numpy as np
from sklearn.linear_model import LinearRegression
```

Load the data

The data is contained in `tempfield.npy` and can be loaded with `np.load(tempfield.npy)`. The first three columns correspond to the `x`, `y`, and `z` locations of the data points, and the 4th column corresponds to the temperature `T` at the respective point. Store the data as you see fit.

In [3]:
```
# YORU CODE GOES HERE
tempfield = np.load('tempfield.npy')
print(tempfield)
```

```
[[ 0  0  0 10]
 [ 8  6  1 15]
 [ 5  2  8 20]
 [ 8  2  6 22]
 [ 5  1  2 16]
 [ 3  3  3 23]
 [ 9  8  2 18]
 [ 3  6  5 19]
 [ 4  6  9 25]
 [ 1  8  2 20]
 [ 1  1  2 28]
 [ 6  4  2 27]]
```

LLS Regression in 3D

Now fit a linear function to the data using the closed for of LLS regression. Use your fitted function to report the predicted temperature at `x = 5, y = 5, z = 5`. You are free to add regularization to your model, but this is not required and will not be graded.

In [8]:
```
# YOUR CODE GOES HERE
x = tempfield[:,0]
y = tempfield[:,1]
z = tempfield[:,2]
Temp = tempfield[:,3]
X = np.vstack([x, y, z, np.ones_like(x)]).T

w = np.linalg.inv(X.T @ X) @ X.T @ Temp.reshape(-1,1)
print("Linear Coefficients:", w.flatten())
X_pred = [5, 5, 5, 1]
Y_pred = X_pred @ w
print("Predicted Temperature is: ", Y_pred)
grad_positive = np.array([5-0.134, 5+0.117, 5+0.726, 1]) @ w
grad_negative = np.array([5+0.134, 5-0.117, 5-0.726, 1]) @ w
print("Temperature when moving towrds the point: ", grad_positive)
print("Temperature when moving opposite to the point: ", grad_negative)
```

```
Linear Coefficients: [-0.13421269  0.11741862  0.72677374 17.83917501]
Predicted Temperature is:  [21.38907338]
Temperature when moving towrds the point:  [21.9484336]
Temperature when moving opposite to the point:  [20.82971317]
```

Gradient Intuition

Using the function you fit in the previous part, in which direction should one move from the point `p = (5,5,5)` to experience the largest decrease in temperature in the immediate neighborhood of the point? Report the specific direction, along with your reasoning.

Your answer goes here

Gradient represents the maximum increase/decrease of a function output. Looking at the code above, the temperature decreases when the next point is in the opposite direction. The gradient of the function is the coeffecients which is (w1, w2, w3) and in this situation it is (-0.134, 0.117, 0.726). So, the direction is opposite of these point which is (0.134, -0.117, -0.726).

Hence, the specific direction for largest decrease is (0.134, -0.117, -0.726).

In [ ]: